AD-A068 368 NAVAL RESEARCH LAB WASHINGTON D C F/G 12/1
A FAST APPROXIMATION TO THE COMPLEMENTARY ERROR FUNCTION FOR US--ETC(U)
APR 79 G W PHILLIPS

UNCLASSIFIED NRL-MR-3963 NL

END
DATE FILMED
6-79
DDC

NRL Memorandum Report 3968

A Fast Approximation to the Complementary Error Function for Use in Fitting Gamma-Ray Peaks

GARY W. PHILLIPS

Radiation Technology Division

12

LEVEL II

April 24, 1979

NAVAL RESEARCH LABORATORY
Washington, D.C.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (14) NRL-MR-
NRL Memorandum Report 3963

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER (9)

4. TITLE (and Subtitle) (6)
A FAST APPROXIMATION TO THE COMPLEMENTARY ERROR FUNCTION FOR USE IN FITTING GAMMA-RAY PEAKS.

5. TYPE OF REPORT & PERIOD COVERED
Final Report on One Phase

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s) (10)
Gary W. Phillips

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Research Laboratory
Washington, DC 20375

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
PE 62712N, Air Task A03S/370D/001B/9F12-100-000
NRL Problem H01-48

11. CONTROLLING OFFICE NAME AND ADDRESS (11) 27
Department of the Navy
Naval Air Systems Command
Washington, DC 20361

12. REPORT DATE
April 1979

13. NUMBER OF PAGES
16

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (12) 16p.

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report) (16) F12 100
Approved for public release; distribution unlimited. (17) WF12 100 000

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

DDC RECEIVED MAY 9 1979 B

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Gamma-ray spectra
Peak fitting
Peak-shape function
Complementary error function
Fast approximation
Computer program
Program HYPERMET

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A fast approximation to the complementary error function has been programmed and tested for use in the peak-shape function for fitting peaks in gamma-ray spectra. The function was compared for speed and accuracy on the NRL ASC 7 computer to the ASC mathematical library version of the complementary error function. The approximation has resulted in a 50% time savings in the computer program HYPERMET which was developed at NRL for automatic analysis of gamma-ray spectra. Source codes for the test programs are given in an Appendix.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

251 950

CONTENTS

A FAST APPROXIMATION TO THE COMPLEMENTARY ERROR FUNCTION FOR USE IN FITTING GAMMA-RAY PEAKS

I. Introduction

The error function and its complement are widely used in fields such as computational physics, numerical methods and statistical analysis, and specifically in fitting gamma-ray peaks for quantitative analysis of spectra from germanium detectors. All of these applications make heavy use of computer calculations. Unfortunately these functions tend to be relatively slow to calculate precisely on a digital computer. In program HYPERMET (1), which was developed at the Naval Research Laboratory (NRL) for automatic peak analysis of gamma-ray spectra, the complementary error function appears in several terms of the peak-shape function which is used in an iterative least-squares fit to the peaks. Timing studies showed that the program was spending up to 80% of its central processing unit (cpu) time in the error-function routine. Since high precision was not required in this application, a search was made for a fast approximation to this function.

II. Computation

The error function *erf*(x) derives its name as the integral of the normal curve of error,

$$erf(x) = (2/\sqrt{\pi})\int_0^x e^{-t^2}\,dt.$$

It is so normalized to have a range of -1 to +1 for $-\infty \le x \le \infty$. The complementary error function is defined by

$$erfc(x) = (2/\sqrt{\pi})\int_x^\infty e^{-t^2}\,dt$$

$$= 1 - erf(x),$$

Note: Manuscript submitted February 13, 1979.

It thus has a range of 2 to 0 for $-\infty \leq x \leq \infty$. On the Texas Instruments ASC Model 7 computer at NRL, the double precision (8-bit exponent, 56-bit mantissa) complementary error function is calculated (2) as follows:

1. For $(0 \leq x \leq 1)$ an 11th order polynomial approximation $P_1(x)$ is used.
2. Within the range $(1 \leq x \leq 2.04)$ various 14th order polynomials $P_2(x)$ are used.
3. Within the range $(2.04 < x \leq 13.306)$ various functions of the form $e^{-x^2}P_3(1/x^2)$ are used, where P_3 is a 14th order polynomial.
4. For $(13.306 < x)$ the function underflows and is set to 0.
5. For $(x < 0)$ the following relation is used.

$$erfc(x) = 2 - erfc(-x)$$

Most non-linear least-squares optimization routines, including that used in HYPERMET, require precise derivatives. For the complementary error function we have

$$\frac{d}{dx}\,(erfc(x)) = -\,(2/\sqrt{\pi})\; e^{-x^2}$$

III. Application

In the peak shape function used by HYPERMET, the main term is a gaussian of width δ and amplitude Γ

$$f_1(x) = \Gamma\; exp(-x^2/\delta^2).$$

Added to this is an exponential "skew" term of amplitude α and slope β on the negative x side of the peak. When folded with a gaussian, representative of random electronic noise, this term can be written as

$$f_2(x) = \alpha\; exp(x/\beta)\; cerf(x - x_o)$$

where the function

$$cerf(x - x_o) = (1/2)\ erfc(x/\delta + \delta/2\beta)$$

has a range of 1 to 0 for $(-\infty \leq x \leq \infty)$ and serves to cut off the exponential part of f_2 smoothly for large x. The second term in the argument of *erfc* introduces an offset by $x_o = -\delta^2/2\beta$. The derivative can be written

$$derf(x - x_o) = \frac{d}{dx}\left[cerf(x - x_o)\right]$$

$$= -(1/\delta\sqrt{\pi})\ exp\left[-(x/\delta + \delta/2\beta)^2\right].$$

The amplitude α of f_2 is usually an order of magnitude less than the amplitude Γ of f_1, and the slope β is usually less than or of the order of the gaussian width δ. When counting statistics are very good, it is sometimes necessary, in order to obtain a good fit, to add a "tail" term f_3, functionally identical to f_2 but with a slope an order of magnitude larger, and a constant "step" term f_4. Both f_3 and f_4 are an order of magnitude less than f_2 in amplitude, and both are also cut off for large x by one-half the complementary error function. Because the offset term in the argument of *erfc* differs for f_1 and f_2 and is zero for f_3, the calculation of the function *cerf* and its derivative *derf* is required up to three times at each data channel (corresponding to x) of the region being fit and for each iteration of the fit.

IV. Approximation

Since the amplitudes of terms f_2, f_3 and f_4 in which the complementary error function appears are small relative to the main gaussian term f_1, since the cutoff term *cerf* is very different from either 1 or 0 only for small absolute x (where f_1 is large), and since the reasons given for inclusion of this term in the peak-shape function are largely empirical, a simpler approximation to *cerf* should perform just as well in the fit. (However, whichever approximation is used, its value and the value of its derivative must be calculated precisely at each channel and

for each iteration in order to avoid cumulative errors during the fitting process.) Several approximations to the error function are given in Ref. (3). The following was chosen as sufficiently accurate and convenient for this purpose: let

$$cutf(x) = (b_1 t + b_2 t^2 + b_3 t^3)\ exp(-x^2)$$

for $x \geq 0$ and $t = 1/(1 + px)$. The constant $p = 0.470\ 47$ and the polynomial coefficients are

$$b_1 = 0.174\ 012\ 1$$

$$b_2 = 0.047\ 939\ 9$$

$$b_3 = 0.373\ 927\ 8.$$

For x less than 0,

$$cutf(x) = 1 - cutf(-x).$$

If we let

$$cerf(x) = cutf(x) + \varepsilon(x),$$

the magnitude of the error term is stated (3) as

$$|\varepsilon(x)| \leq 1.25 \times 10^{-5}.$$

The exact derivative is given by

$$dutf(x) = \frac{d}{dx}\ (cutf(x))$$

$$= -\ (2x)\ cutf(x)$$

$$-\ pt^2(b_1 + 2tb_2 + 3t^2 b_3)\ exp(-x^2).$$

V. Tests

The functions *cutf* and *dutf* were tested on the ASC 7 at NRL for speed and accuracy against the functions *cerf* and *derf*. The ASC is a two-pipe-line vector-oriented machine which in an overlapped fasion can retrieve arrays of data, perform simple algebraic manipulations on the arrays, and store the results. For arrays of length ≥ 10, this is usually much more efficient than the corresponding scalar operations on the data element by element. Many of the standard mathematical functions have both vector and scalar routines available for their calculation, and the FORTRAN compiler is written so as to automatically invoke the vector routines when applicable for processing expressions involving arrays in the source code. Unfortunately, the complementary error function is available only as a scalar routine.

For this test two routines (4) were written in FORTRAN to calculate *cutf* and *dutf* for x varying between -5.0 and +4.9 in 100 steps of 0.1 each, which covers a range typical for program HYPERMET. One routine was written to store intermediate results in arrays in order to compile efficiently in vector code. The second routine was written to compile efficiently in scalar code. A third program was written to calculate *cerf* and *derf* for the same steps over the same range. Then the lower and upper limits of the range of calculation were each incremented 100 times in steps of 0.01 and the calculations repeated, for a total of 10,000 calculations of each function. Finally, the above calculations were repeated ten times for a grand total of 100,000 calculations of each function. The source code was compiled and executed in double precision at three different optional levels of compiler optimization (5),

1. I-level, scalar code with direct translation of source code, no optimization
2. J-level, scalar code with local and global optimization
3. K-level, vector code with local and global optimization.

VI. Accuracy

In Figs. 1 and 2 are plotted the results of the calculation for *cutf* and *dutf* and the errors

$$epsc(x) = cerf(x) - cutf(x)$$

$$epsd(x) = derf(x) - dutf(x)$$

between x = 0.0 and 5.0.

For negative x, *cutf*(x) = 1.0 - *cutf*(-x), while the other curves are symmetric about x=0. Over the range of $0 \le x \le 1.65$, the error curve *epsc* oscillates between $\pm 1.10 \times 10^{-5}$. The sign of the error is indicated in parentheses and changes at each cusp in the curve as the error passes through zero. For x > 1.65, *epsc* declines gradually with *cutf*. At x=5, *cutf* = 7.99×10^{-13} and $|epsc| = 3.08 \times 10^{-14}$ or about 4%. Similar behavior is observed for the error *epsd* in the derivative *dutf*.

VII. Timing

Table 1 gives the cpu execution times for 100,000 fast-approximation calculations of *cutf* and *dutf* in both the scalar- and vector-coded versions compared to the times for the full calculation of *cerf* and *derf*, all codes compiled and executed in double precision on the Texas Instruments ASC 7 at NRL. Results are given for three different compiler levels I, J, and K as described previously. Table 1 shows significant savings at all compiler levels, with a maximum savings at the K compiler level for the vector-coded version of the fast approximation, which executes in 23% of the time required for the full calculation. At the I and J levels, maximum savings are found for the scalar-coded version, which executes in about 60% of the time required for the full calculation.

In Table 2 the lines of code needed to calculate the derivatives were deleted before compilation. Here the savings are similar at the K level but smaller at the I and J levels. In fact, the vector-coded version at the I level requires about 13% longer time to execute than does the full calculation. This is somewhat surprising and apparently represents the economy of an optimized assembly language code versus an optimized FORTRAN-compiled code. For the scalar code, this seems to indicate that large savings can be realized by coding frequently used functions in assembly language rather than FORTRAN.

VIII. Results

In program HYPERMET, subroutine FUNC which calculates the peak-shape function has been rewritten in order to produce efficient vector code at the K-level of compiler optimization. This change resulted in a reduction in execution time to about 40% of that required for the previous version. The vector-coded fast approximation for the functions *cutf* and *dutf* was then added in-line at each of three locations, at the code for calculating f_1, f_2 and f_3 where the functions *cerf* and *derf* had been used previously. With this change another factor of two in execution speed was achieved. As the final result of these changes, the program HYPERMET now executes on the ASC 7 in about one-fifth the time required by the previous version to analyze a typical gamma-ray spectrum.

IX. References

1. G.W. Phillips and K.W. Marlow, Nucl. Instrum. Methods 137 (1976) 525; NRL Memorandum Report 3198, January 1976.
2. "ASC Mathematical Library," Manual No. 929978-2, Texas Instruments, Austin, Texas, January 1977.
3. "Handbook of Mathematical Functions," Dover Publications, New York, 1965.
4. See Appendix for source listings.
5. "ASC FORTRAN Reference Manual," Manual No. 930044-3, Texas Instruments, Austin, Texas, January 1978.

Table 1.

Complementary Error Function and Derivatives,
Time (sec.) for 100,000 Calculations

	Fast Approximation		Full Calculation
Compiler Level	Vector Code	Scalar Code	Scalar Code
I	7.23	5.63	8.82
J	5.50	5.04	8.49
K	1.35	2.86	5.95

Table 2.

Complementary Error Function Only
Time (sec.) for 100,000 Calculations

	Fast Approximation		Full Calculation
Compiler Level	Vector Code	Scalar Code	Scalar Code
I	5.94	4.85	5.23
J	4.66	4.41	5.10
K	1.19	1.40	5.02

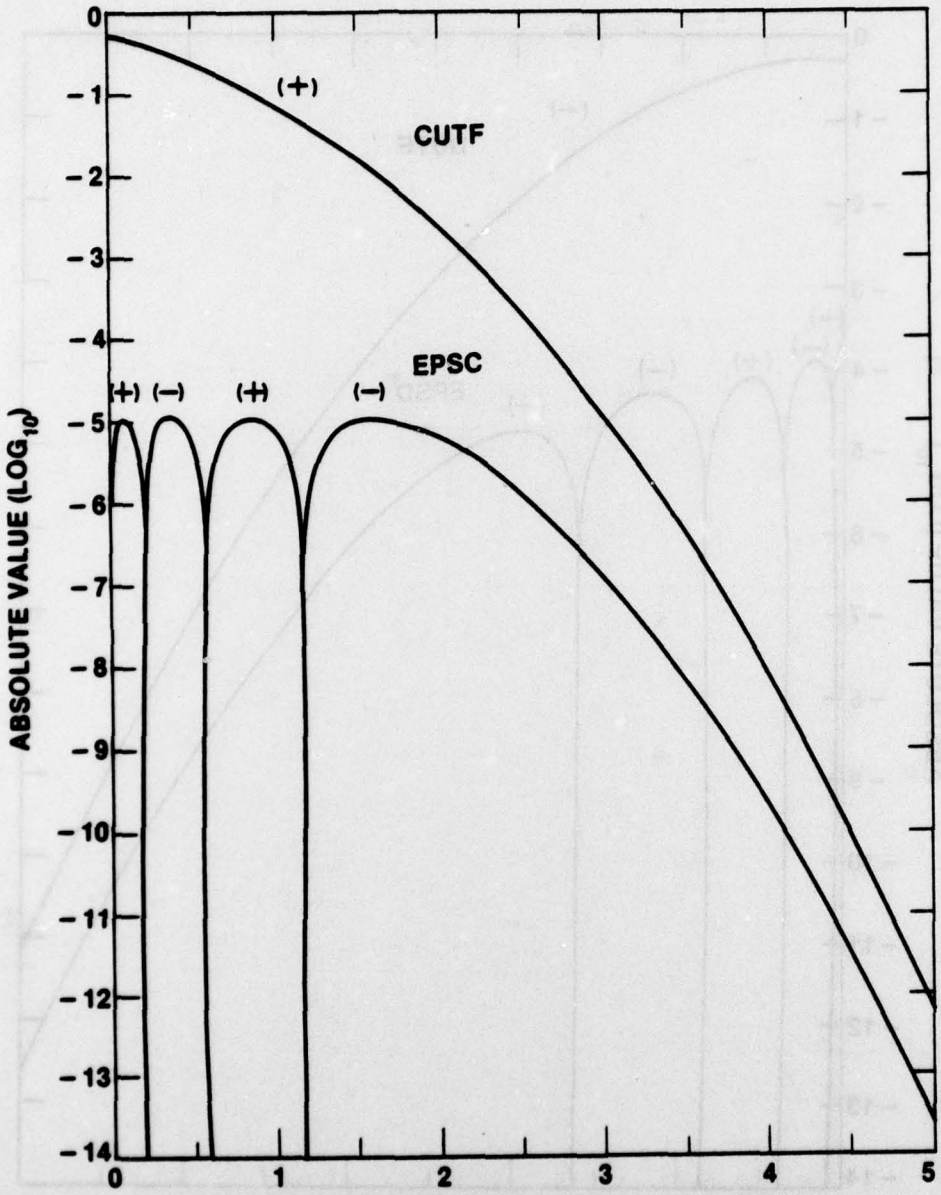

Fig. 1 - Plot of the fast approximation CUTF to the complementary error function, and the error EPSC, as defined in the text. Plotted is the $\log_{10}$ of the absolute value of the functions. The sign is indicated in parentheses above the curve. The cusps in the curve for EPSC result from sign changes as the function passes through zero.

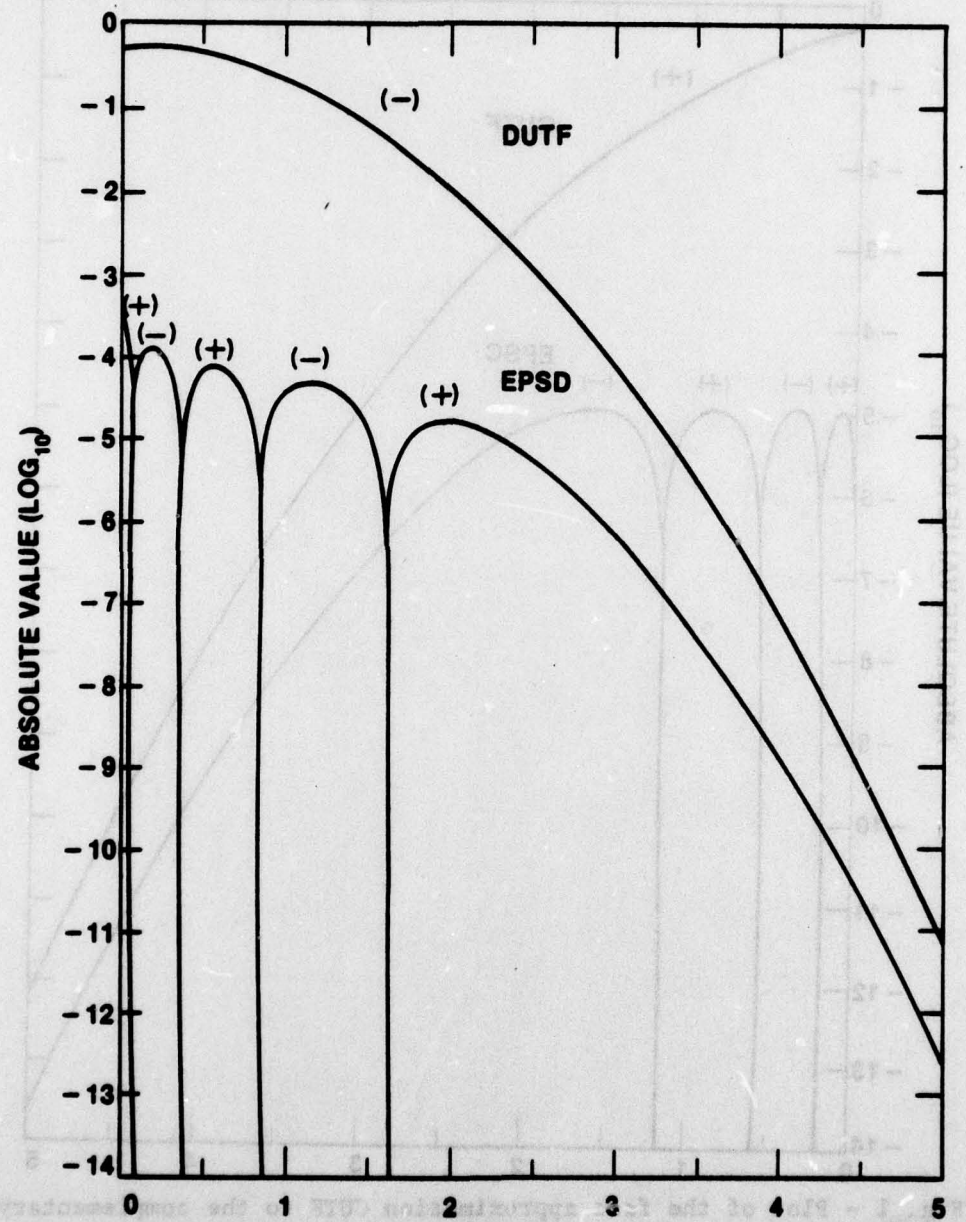

Fig. 2 - Plot of the derivative DUTF of the fast approximation to the complementary error function CUTF, and the error EPSD as defined in the text. The sign is indicated in parentheses above the curve. The cusps in the curve for EPSD result from sign changes as the function passes through zero.

APPENDIX

```
      PROGRAM CUTFV
C     PROGRAM FOR TIMING TESTS OF A RATIONAL APPROXIMATION TO THE
C     COMPLEMENTARY ERROR FUNCTION CUTF AND ITS DERIVATIVE DUTF
C     VECTOR CODED VERSION
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION GAUSN(100),T3(100),T1(100),T2(100),U(100)
      DIMENSION CUTF(100),DUTF(100)
      DIMENSION FT(100),DFT(100),DFX(100)
      DATA A1/0.1740121D00/,A2/-0.0479399D00/
      DATA A3/0.3739278D00/,AP/0.47047D00/
C
      N=100
      DU=0.01D00
      DO 1000 KX=1,10
      DO 1000 IX=1,100
         UO=-5.11D00+DU*DFLOAT(IX)
         DO 100 I=1,N
            U(I)=UO+0.1D00*DFLOAT(I)
            GAUSN(I)=DEXP(-U(I)**2)
  100    CONTINUE
         K=IDINT(-UO*10.)
         DO 200 I=1,K
            U(I)=-U(I)
  200    CONTINUE
         DO 300 I=1,N
            T1(I)=1.D00/(1.D00+AP*U(I))
            T2(I)=T1(I)*T1(I)
            T3(I)=T1(I)*T2(I)
            FT(I)=A1*T1(I)+A2*T2(I)+A3*T3(I)
            DFT(I)=A1+2.D00*A2*T1(I)+3.D00*A3*T2(I)
            DFX(I)=-DFT(I)*AP*T2(I)
  300    CONTINUE
         DO 400 I=1,N
            CUTF(I)=FT(I)*GAUSN(I)
            DUTF(I)=DFX(I)*GAUSN(I)-2.D00*U(I)*CUTF(I)
  400    CONTINUE
         DO 500 I=1,K
  500       CUTF(I)=1.D00-CUTF(I)
 1000 CONTINUE
C
      STOP
      END
```

```
      PROGRAM CUTFS
C     PROGRAM FOR TIMING TESTS OF A RATIONAL APPROXIMATION TO THE
C     COMPLEMENTARY ERROR FUNCTION CUTF AND ITS DERIVATIVE DUTF
C     SCALAR CODED VERSION
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION U(100)
      DIMENSION CUTF(100),DUTF(100)
      DATA A1/0.1740121D00/,A2/-0.0479399D00/
      DATA A3/0.3739278D00/,AP/0.47047D00/
C
      N=100
      DU=0.01D00
      DO 1000 KX=1,10
      DO 1000 IX=1,100
         UO=-5.11D00+DU*DFLOAT(IX)
         DO 100 I=1,N
            U(I)=UO+0.1D00*DFLOAT(I)
            UA=DABS(U(I))
            T=1.D00/(1.D00+AP*UA)
            HN=DEXP(-U(I)**2)
            CUTF(I)=HN*T*(A1+T*(A2+T*A3))
            DUTF(I)=-HN*AP*T*T*(A1+T*(2.D00*A2+T*3.D00*A3))
     *              -2.D00*UA*CUTF(I)
 100     CONTINUE
         K=IDINT(U-UO*10.)
         DO 200 I=1,K
 200        CUTF(I)=1.D00-CUTF(I)
 1000 CONTINUE
C
      STOP
      END
```

```
      PROGRAM CERF
C     PROGRAM FOR TIMING TESTS OF THE ASC LIBRARY ROUTINE FOR THE
C     COMPLEMENTARY ERROR FUNCTION CERF AND ITS DERIVATIVE DERF
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION U(100)
      DIMENSION CERF(100),DERF(100)
C
      RPI=0.56418935D00
      N=100
      DU=0.01D00
      DO 1000 KX=1,10
      DO 1000 IX=1,100
         UO=-5.11D00+DU*DFLOAT(IX)
         DO 100 I=1,N
            U(I)=UO+0.1D00*DFLOAT(I)
            CERF(I)=0.5D00*DERFC(U(I))
            DERF(I)=-RPI*DEXP(-U(I)**2)
 100     CONTINUE
 1000 CONTINUE
C
      STOP
      END
```